README

C64Net WiFi Firmware 3.5.4 (C)2016-2020 Bo Zimmerman
Please read the LICENSE file for license information
Please read the NOTICE file for credits information

Intro:

This firmware code, "C64Net WiFi Firmware", provides an api for communication between a serial terminal and the ESP8266 or ESP32. It simulates the old-style Hayes "AT" commands, so that the ESP8266 appears to the terminal as if it were a Hayes modem. These "AT" commands may then be issued to the ESP8266 in order to force it to connect to a wireless access point, and from there to make one or more manageable connections to the internet. It also includes a streaming serial "telnet" mode, and server/port listener capabilities.

The default baud rate that the firmware establishes is 1200. Be sure to use the commands below to set this to the proper baud rate for your terminal/host computer. If you plan on using the Commodore 8-bit utilities, keep the firmware baud rate saved at 1200.

Building:

To build this firmware for the ESP-01 or ESP-12, I used the Arduino IDE rev1.8.10 with the latest Arduino ESP8266 libraries using the Generic ESP8266 Module. Install the libraries from the Boards Manager using http://arduino.esp8266.com/stable/package_esp8266com_index.json . Both versions of the firmware are built with SPIFFS enabled. On the ESP8266, I used settings of 1M/160k, but you may want to alter that depending on your hardware and needs.

To build this firmware for the ESP-32, I also used the Arduino IDE rev 1.8.10 with the Arduino ESP32 Dev Module installed from the board manager * , with the following settings: QIO, 4MB, 80MHZ, 921600, NONE, Avrisp Mk II **. I found that baud changing corrupts the serial bus as of 1/13/2018, so the following changes were made to the base libraries:
> Added to esp32-hal-uart.c:

```
void uartChangeBaudRate(uint8_t uart_nr, uint32_t baudrate)
{
    uartSetBaudRate(&_uart_bus_array[uart_nr], baudrate);
}
void uartChangeConfig(uint8_t uart_nr, uint32_t config)
{
    uart_t* uart = &_uart_bus_array[uart_nr];
    UART_MUTEX_LOCK();
    uart->dev->conf0.val = config;
    #define TWO_STOP_BITS_CONF 0x3
    #define ONE_STOP_BITS_CONF 0x1

    if ( uart->dev->conf0.stop_bit_num == TWO_STOP_BITS_CONF) {
        uart->dev->conf0.stop_bit_num = ONE_STOP_BITS_CONF;
        uart->dev->rs485_conf.dl1_en = 1;
    }
    UART_MUTEX_UNLOCK();
}
size_t uartWritesRemaining(uart_t* uart)
{
    if(uart == NULL) {
        return 0;
    }
    size_t remain;
    UART_MUTEX_LOCK();
    remain = 0x7F - uart->dev->status.txfifo_cnt;
    UART_MUTEX_UNLOCK();
    return remain;
}
> Added to esp32-hal-uart.h:
void uartChangeBaudRate(uint8_t uart_nr, uint32_t baudrate);
void uartChangeConfig(uint8_t uart_nr, uint32_t config);
size_t uartWritesRemaining(uart_t* uart);
> Added to HardwareSerial.cpp:
void HardwareSerial::changeBaudRate(int baudRate) { uartChangeBaudRate(_uart_nr,
baudRate); }
void HardwareSerial::changeConfig(uint32_t config) { uartChangeConfig(_uart_nr,
config); }
> *Changes* to HardwareSerial.cpp
  In the function: void HardwareSerial::begin(unsigned long baud, uint32_t
config, int8_t rxPin, int8_t txPin, bool invert)
  change: _uart = uartBegin(_uart_nr, baud, config, rxPin, txPin, 256, invert);
  to    : _uart = uartBegin(_uart_nr, baud, config, rxPin, txPin, 4096, invert);
> Added to HardwareSerial.h:
void changeBaudRate(int baudRate);
void changeConfig(uint32_t config);
```

* On the ESP32, if you want to build firmware that is Update compatible with the official production zimodem, you'll need to install the exact libraries I use from http://www.zimmers.net/otherprojs/esp32_v0.zip . If you don't care about that, using the latest libs work just fine.

Using:

Upon initialization, or any time the ESP module is reset, the modem will display its version, and some information about the host hardware, and then read a configuration file from the internal SPIFFS to re-establish the previously set baud rate, and to attempt to re-connect to the previously connected wireless router. The first time it is run, the firmware will set a baud rate of 1200 and display INITALIZED to let you know that no previous wifi configuration was found. Once the serial terminal displays READY, it is ready to receive commands.

The first command you'll probably want to enter is AT+CONFIG to connect to a wireless router, and set your flow control and other command mode settings.

Afterwards, if you plan to use this primarily from a computer that doesn't need linefeeds, such as the C64, you'll want to enter ATR0 to go into carriage-return only mode, and then at&w to save this setting as well.

If you want to operate at a higher baud, you'll want to enter ATB9600 (or whatever baud rate you want to try), and then reconnect your terminal program to the modem at that new baud rate. If everything looks good, and you want to keep the new baud rate across restarts, save the new baud rate with AT&W. Warning though: Most of the example C64 programs assume the modem defaults to 1200 baud.

If you want to connect to a remote telnet server, eg coffeemud.net, port 23, you'll want to enter ATDT"coffeemud.net:23". Don't forget to set your terminal program to the proper translation mode (ANSI, ASCII, or whatever).

If you are using a Commodore Graphics terminal program and want to connect to a Commodore BBS, eg cottonwoodbbs.dyndns.org port 6502, you'll want to enter ATD"cottonwoodbbs.dyndns.org:6502".

If you want to use Q-Link, you need to add a phone number alias first. To do this, enter ATP"5551212=q-link.net:5190" or enter it from the config menu AT+CONFIG. From the C64 Q-Link client, select "Hayes compatible" 1200 baud modem when prompted.

If you want to run a Commodore BBS program using the modem, you'll want to configure the BBS program to the same idle baud rate that your modem is using (1200 baud by default), configure it for a Hayes style modem (or the C=1670), and either create a persistant listener using AT+CONFIG, or use an initialization string of "ATR0E0S0=1S41=1A6400" plus any other recommended settings from the BBS program. This creates a listener at port 6400 that switches directly to stream mode on the first ring, with no linefeed carriage returns, and no keystroke echo. Your BBS program may require you add certain other settings, such as V0 or X1.. which you should also do.

If you want to try printing to a CUPS/IPP-printer, enter AT+PRINT?:<host>:<port>/<path>, followed by your data. Where ? is A)scii, P)etscii, or R)aw. A 5 second pause in incoming data completes the document and returns to command mode. Example: AT+PRINTR:192.168.1.10:631/ipp/printer -- followed by ENTER and then the data to print, without any pause longer than 5 seconds. Subsequent to doing this, using AT+PRINT will repeat the previous URL.

ESP32 Guru Modem users with SD-cards can enter AT+SHELL to get a shell command prompt. Enter ? to get a list of shell commands.

Command Set:

The command set is as follows (not case sensitive):

ATZ : closes all open socket connections (preserving the Access Point connection), stops all listeners, and resets the state of the Command processor to the saved configuration, preserving the current baud rate and wifi connection.

A/ : Repeats the previous command

ATI : re-shows the startup message, including wifi connection information.
ATI0 : same as ATI
ATI1 : Shows the current common variable settings, common 'S' registers.
ATI2 : Shows the modem's current IP address
ATI3 : Shows the modem's current Wireless Router connection
ATI4 : Shows only the firmware current version
ATI5 : Shows all the current variable settings, all 'S' registers.
ATI6 : Shows the current mac address.
ATI7 : Shows the current formatted time (see AT&T).
ATI8 : Shows the firmware build date/time

ATA : If a server listener has generated a RING, then ATA will switch the last rung connection to Stream mode (see ATD).

ATAn : Causes the modem to create a server listening on port n. When a connection is received, the terminal will generate 1 or more RINGs according to the ATS0 register, followed by a normal CONNECT respose. At this point, all other commands related to connections may be used normally, unless ATS41 is > 0, in which case incoming connections are automatically sent to Stream mode as per ATD or ATA. Listeners are listed along with other connections using ATC0.
ATAPn : Adding a P modifier causes all incoming connection input to be translated to PETSCII
ATATn : Adding a T modifier causes connection streaming input to be translated per TELNET when the changed to Stream mode
ATAEn : Adding a E modifier causes connection terminal echo to be enabled when the changed to Stream mode
ATAXn : Adding a X modifier causes connection XON/XOFF flow control to be enabled when the changed to Stream mode.

ATN0 : Shuts down all listeners, leaving client connections open
ATNn : if n > 0 then same as ATAn

```
ATE0 : Turns serial terminal echo off for command mode.
ATE1 : Turns serial terminal echo on for command mode.

ATV0 : Turns off verbose responses mode (Uses Terse Numeric response mode)
ATV1 : Turns on verbose responses mode (Uses Word response mode)

ATX0 : Turns off extended response codes (1/CONNECT instead of 5/CONNECT 2,
       etc..)
ATX1 : Turns on extended response codes (5/CONNECT 2 instead of 1/CONNECT,
       etc..)

ATF0 : Turns on rts/cts flow control.
ATF1 : Turns on xon/xoff flow control.
ATF2 : Turns on xon/xoff flow control, sets XON mode (if necessary), and, in
       command mode, will immediately go to XOFF when a single connection packet
       is received. This is very useful when the client wants to ensure it only
       receives one packet to process.  You can think of this as an alternative
       way to use xon/xoff by having XOFF automatic between packets.
ATF3 : Similar to ATF2 except that the default is XOFF, and, in command mode, a
       XON code from the user will immediately trigger either an empty packet
       response [ 0 0 0 ], or a real packet if one is available.  After this, as
       in ATF2, XOFF is automatically set.
ATF4 : Turns off flow control for command mode

ATQ0 : Turns off quiet mode (Sends response codes)
ATQ1 : Turns on quiet mode (Stops sending response codes)

ATR0 : Suppresses linefeed (\n $0a) in end of lines.  Will only send carriage
       return (\r $0d).
ATR1 : Sends \r\n ($0d0a) as end of line string.
ATR2 : Sends \n\r ($0a0d) as end of line string.
ATR3 : Suppresses carriage return (\r $0d) in end of lines.  Will only send
       linefeed (\n $0a).

ATBn : Sets a new serial Baud Rate. Takes effect immediately.
ATB"n,xYz" : Sets baud rate n, bits x, parity (E,O,M, or N) for Y, and stop bits
       z.

ATW : List all wireless network access points scanned within range.  The
      response for each entry is the SSID, following by the RSSI, followed by an
      * character is the connection is encrypted.
ATWn : Where n > 0, this lists up to n wireless network access points scanned
       within range.  The response for each entry is the SSID, following by the
       RSSI, followed by an * character is the connection is encrypted.

ATW"[SSI],[PASSWORD]" : Connects to the wireless access point with the given
       SSI, using the given password.
ATWP : Adding a P modifier is the same as all forms of ATW, with both arguments
       and results presented in PETSCII.

ATD : Start a streaming connection between the current opened connection.  Use
      "+++" to exit back to Command mode.
ATDn : Where n > 0, this will start a streaming connection between the
       previously opened connection with an id the same as n.  Use "+++" to exit
       back to Command mode.
ATD"[HOSTNAME]:[PORT]" : This opens a streaming connection between the terminal
       and the given host/port. Use "+++" to disconnect and exit back to command
       mode.
ATDP"[HOSTNAME]:[PORT]" : Adding a P modifier causes connection input to be
       translated to PETSCII during the streaming session.
ATDT"[HOSTNAME]:[PORT]" : Adding a T modifier causes connection input to be
       translated per TELNET during the streaming session.
ATDE"[HOSTNAME]:[PORT]" : Adding a E modifier causes terminal echo to be enabled
```

```
      that streaming session.
ATDX"[HOSTNAME]:[PORT]" : Adding a X modifier causes XON/XOFF flow control to be
      enabled that streaming session.
ATDnnnnnnn : Where n=0-9, if the digits exist in the phonebook (see ATP), it
      will try connect to that host, with those modifiers, from the phonebook.

ATC : Shows information about the current network connection in the following
      format "[CONNECTION STATE] [CONNECTION ID] [CONNECTED TO HOST]:[CONNECTED
      TO PORT]"
ATC0 : Lists information about all of the network connections in the following
       format "[CONNECTION STATE] [CONNECTION ID] [CONNECTED TO HOST]:[CONNECTED
       TO PORT]", including any Server (ATA) listeners.
ATCn : Where n > 0, this changes the Current connection to the one with the
       given ID.  If no connection exists with the given id, ERROR is returned.
ATC"[HOSTNAME]:[PORT]" : Creates a new connection to the given host and port,
       assigning a new id if the connection is successful, and making this
       connection the new Current connection.  The quotes and colon are
       required.
ATCP"[HOSTNAME]:[PORT]" : Adding a P modifier causes all connection input to be
       translated to PETSCII
ATCT"[HOSTNAME]:[PORT]" : Adding a T modifier causes streaming input to be
       translated per TELNET when the changed to Stream mode
ATCE"[HOSTNAME]:[PORT]" : Adding a E modifier causes terminal echo to be enabled
        when the changed to Stream mode
ATCX"[HOSTNAME]:[PORT]" : Adding a X modifier causes XON/XOFF flow control to be
        enabled  when the changed to Stream mode

ATH  : Hangs up (disconnects and deletes) all open connections.  Does not close
       Server listeners.
ATH0 : Hangs up (disconnects and deletes) the current opened connection.
ATHn : Hangs up (disconnects and deletes) the open connection with the given id.
       Closing a Server (ATA) listener does not close any connections received
       from that listener.

ATO : Re-enters a Streaming session (see ATD) under the previous settings, with
      the current (previous) connection.

ATP : Lists all existing phonebook entries, with the format phone number
      followed by ATD modifiers, followed by the host and port
ATP"[NUMBER]=[HOSTNAME]:[PORT]" : Adds or Modifies an entry to the phonebook
      with the given 7 digit number, host, and port. Use ATDnnnnn.. to connect.
ATPP"[NUMBER]=[HOSTNAME]:[PORT]" : Adding a P modifier causes connection input
      to be translated to PETSCII  when connected to that entry.
ATPT"[NUMBER]=[HOSTNAME]:[PORT]" : Adding a T modifier causes connection input
      to be translated per TELNET  when connected to that entry.
ATPE"[NUMBER]=[HOSTNAME]:[PORT]" : Adding a E modifier causes terminal echo to
      be enabled when connected to that entry.
ATPX"[NUMBER]=[HOSTNAME]:[PORT]" : Adding a X modifier causes XON/XOFF flow
      control to be enabled when connected to that entry.
ATP"[NUMBER]=DELETE" : Removes the phonebook entry with the given number.

ATS0=n : Changes the number of RING messages received before a CONNECT response
         is sent, on incoming Server listeners.
ATS1=n : Unimplemented, always returns OK
ATS2=n : Change the escape character (n = 0-255), Defaults to ASCII decimal 43
         ("+")
ATS3=n : Change the Carriage Return Character   (n = 0-127), Defaults to ASCII
         decimal 13 (Carriage Return)
ATS4=n : Change the Line Feed Character (0-127), Defaults ASCII decimal 10 (Line
         Feed)
ATS5=n : Change the Backspace Character (0-32), ASCII decimal 8 (Backspace)
ATS6 ... 39=n : Unimplemented, always returns OK
ATS40=n : Change the size of the connection packets (n > 0), Defaults to 127
          bytes
```

```
ATS41=n : When n > 0, all incoming Server listener connections are immediately
          sent to Stream mode.  If n=0, connections remain in normal command
          mode (default).
ATS42=n : Set the CRC8 for an attached Transmit command.  e.g.
ATS42=123T"[MESSAGE]" returns error unless 123 is CRC8 of "[MESSAGE]".
ATS43=n : Sets a standby baud rate n for the next incoming or outgoing
          connection only.  ATZ clears.
ATS44=n : Sets an automatic delay of n milliseconds after most bytes written to
          the Serial port.  This is for computers that support a baud rate, but
          can't really keep up, and you don't want to use flow control.
ATS45=n : Changes how packet and at&g data is delivered.  0 is normal binary
          with normal headers, 1 is 78 char HEX digit streams followed by EOLN
          with hex digit headers, 2 is decimal digits followed by EOLN, with
          decimal digit headers, 3 is normal without SUM header.
ATS46=n : Changes DCD status. n=0 is default DCD=HIGH=online. n=1 is
          DCD=LOW=online. n=2 always HIGH. n=3 always LOW.
ATS47=n : Changes DCD pin number, n=2 is default
ATS48=n : Changes CTS status. n=0 is default CTS=HIGH=active. n=1 is
          CTS=LOW=active. n=2 always HIGH. n=3 always LOW.
ATS49=n : Changes CTS pin number, n=0 is default on ESP01, and default is 5
          otherwise
ATS50=n : Changes RTS status. n=0 is default RTS=HIGH=active. n=1 is
          RTS=LOW=active. n=2 always HIGH. n=3 always LOW. (N/A on ESP01)
ATS51=n : Changes RTS pin number, n=4 is default (N/A on ESP01)
ATS52=n : Changes RI status. n=0 is default RI=HIGH=active. n=1 is
          RTS=LOW=active. n=2 always HIGH. n=3 always LOW. (N/A on ESP01)
ATS53=n : Changes RI pin number, n=14 is default (N/A on ESP01)
ATS54=n : Changes DTR status. n=0 is default DTR=HIGH=active. n=1 is
          RTS=LOW=active. n=2 always HIGH. n=3 always LOW. (N/A on ESP01)
ATS55=n : Changes DTR pin number, n=12 is default (N/A on ESP01)
ATS56=n : Changes DSR status. n=0 is default DSR=HIGH=active. n=1 is
          RTS=LOW=active. n=2 always HIGH. n=3 always LOW. (N/A on ESP01)
ATS57=n : Changes DSR pin number, n=13 is default (N/A on ESP01)
ATS60=n : When n > 0, immediately saves existing listeners and automatically
          restores them later. n=0 to clear.
ATS61=n : When n > 0, sets the number of seconds to timeout a print job stream
          (AT+PRINT). Default is 5 seconds

+++ : With a 1 second pause with no other characters afterwards, this will
      disconnect the current opened connection.

ATT"[MESSAGE]" : Transmit the given text string, with \r\n at the end, on the
      current connection.
ATTn : Where n > 0, this starts a transmit of exactly n bytes to the current
       connection.  The \n from entering this command must be followed by the n
       bytes to transmit.
ATTP"[MESSAGE]" : Transmit the given text string, translating petscii to ascii,
       with \r\n at the end, on the current connection.
ATTPn : Where n > 0, this starts a transmit of exactly n bytes to the current
        connection, translating petscii to ascii.  The \n from entering this
        command must be followed by the n bytes to transmit.
ATT+"[MESSAGE] : A + argument may be used to force the 'T' command to return the
        CRC8 of the message instead of OK, when successful.

ATL0  : Re-sends the most recently sent data packet again
ATLn  : Re-sends the most recently sent data packet for connection id n.

AT&H  : Shows a help file from the web, or brief help otherwise.  Use &H6502 to
        reiforce web download.
AT&L  : Reloads the saved configuration.
AT&W  : Saves the current configuration: WiFi settings(ATW), baud rate (ATB),
        end of line (ATR) settings, flow control (ATF), echo mode (ATE),
        extended responses (ATX), verbose responses (ATV), quiet responses
        (ATQ), PETSCII mode (AT&P1), pin statuses (ATS46 - S58), Rings (ATS0),
        Listener Stream-mode (ATS41), and Listener restore (ATS60), printer spec
        (AT+PRINT)
AT&F  : Restores the modem to factory default settings.  Use &F86 to reformat
        the SPIFFS.
AT&On : n is 1 to turn on internal serial-reception log, n is 0 to turn off or
        view a previously turned-on log.
AT&U  : Checks the firmware home page to see if a new version is available.
AT&U6502 : Will update the firmware from the home page on the web.
AT&U=x: Will update the firmware from the web to custom version x.
AT&Kn : Flow Control, similar to ATFn, n=0,1,2: disable, n=3,6: rts/cts, n=4,5:
        Xon/Xoff
AT&Pn : Where n > 0, all command mode input and output will be translated
        to/from PETSCII before internal processing.  This will not affect
        received packet data, or the stream mode.
AT&Nx : Shows the status of ESP module I/O pin x
AT&Mn : Adds the byte denoted by n to a list of mask-out bytes.  These are bytes
        that are not transmitted to the serial port in command mode incoming
        packets.  If this command is followed by a C, N, or A command on the
        SAME LINE, then the setting will apply ONLY to that connection or
        listener.
AT&M  : Resets the mask-out bytes list. No bytes will be masked-out. If this
        command is followed by a C, N, or A command on the SAME LINE, then the
        setting will apply ONLY to that connection or listener.
AT&Dn : Adds the byte denoted by n to a list of delimiter bytes.  These are
        bytes that will compose the last byte in a command-mode incoming packet
        that is still shorter than the limit set by ATS40. This is useful for
        CR-LF formatted data.  If this command is followed by a C, N, or A
        command on the SAME LINE, then the setting will apply ONLY to that
        connection or listener.
AT&D  : Resets the delimiter bytes list. No bytes will be delimited, and packets
        will contain as many bytes as are received and allowed by ATS40.  If
        this command is followed by a C, N, or A command on the SAME LINE, then
        the setting will apply ONLY to that connection or listener.

AT&S"40=[HOSTNAME]" : Change the modem hostname
AT&S"41=[TERMTYPE]" : Change the telnet 'termtype' response string

AT&T"[TIMEZONE],[TIME FORMAT],[NTP URL]" : set up the NTP clock. DISABLE to
        disable. Format is like Java SimpleDateFormat, but with % escapes. Each
        argument is optional.
AT&G"[HOSTNAME]:[PORT]/[FILENAME]" : Streams a file from an HTTP source on the
        internet.  The header contains channel 0, file length, and an 8-bit sum
        of all the bytes in the forthcoming file, followed by the bytes of the
        file, all formatted as a normal packet.  An ASCII 3 (CNTRL-C) received
        during the transfer will abort. The S44 register can be used to create
        artificial delays in this output.  XON/XOFF Flow control also remains in
        effect with, on a byte-by-byte basis for the auto and manual flow
        control systems. Requires flash space for caching, or S45=3 to eliminate
        the SUM header.
AT&Y  : Resets the state machine string. No state machine will be executed.
AT&Yn : Change the current state (for command mode AND current connection) to
        state n, where n is a decimal number.
AT&Y"[CODED STATE MACHINE]" : Adds the coded format string to a state machine.
        If this command is followed by a C, N, or A command on the SAME LINE,
        then the setting will apply ONLY to that connection or listener.   State
Machine Format: MMcCCNN ...  States are numbered by their order in the
                list starting with 00.  Non-matches automatically go to
                the next state until a match is made.   'MM' is hex byte
                to match (or 00 to match all).  'c' is one of these
commands :e=eat byte, p=push byte to que, d=send byte, q=send all queued,
          x=flush queue, r=replace with byte represented by hex CC.  'C' is
          either '-', one of the command letters above, or a hex byte value if
          the first command was 'r'.  'NN' is the next state to go to, with 00
          being the first state.
```